



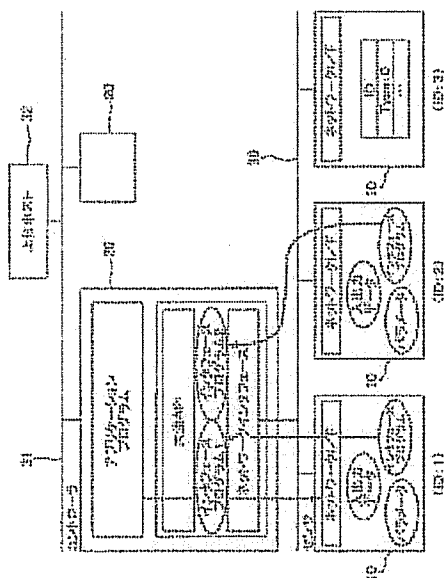
**SENSOR, CONTROLLER AND SENSOR SYSTEM****Publication number:** JP2000270384 (A)**Publication date:** 2000-09-29**Inventor(s):** MIYATA YOSHIAKI; KADOWAKI MASANORI**Applicant(s):** OMRON TATEISI ELECTRONICS CO**Classification:**

- international: H04Q9/00; G06F9/00; G06F9/44; G06F9/445; H04L12/40;  
H04Q9/00; G06F9/00; G06F9/44; G06F9/445; H04L12/40;  
(IPC1-7): H04Q9/00; H04L12/40

- European: G06F9/445B5

**Application number:** JP19990066343 19990312**Priority number(s):** JP19990066343 19990312; US20010966591 20010927**Also published as:** US2003061403 (A1) US6895572 (B2)**Abstract of JP 2000270384 (A)**

**PROBLEM TO BE SOLVED:** To provide a sensor system where information specific to a sensor for communication between the sensor and a controller can be set to the controller. **SOLUTION:** A sensor 10 stores its own interface program and gives the interface program to a controller 20 via a network 30. The controller up-loads the interface program for the sensor received from the sensor and accesses the sensor in an optimum operation environment by using the acquired interface program. However, since the interface program has only to be downloaded via the network, it can simply be executed.



Data supplied from the esp@cenet database — Worldwide

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2000-270384

(P2000-270384A)

(43) 公開日 平成12年9月29日 (2000.9.29)

(51) Int.Cl. <sup>7</sup>	識別記号	F I	データベース* (参考)
H 0 4 Q 9/00	3 1 1	H 0 4 Q 9/00	3 1 1 P 5 K 0 3 2
H 0 4 L 12/40		H 0 4 L 11/00	3 2 0 5 K 0 4 8

審査請求 未請求 請求項の数3 O L (全 13 頁)

(21) 出願番号 特願平11-66343

(22) 出願日 平成11年3月12日 (1999.3.12)

(71) 出願人 000002945

オムロン株式会社

京都府京都市右京区花園土堂町10番地

(72) 発明者 宮田 佳昭

京都府京都市右京区花園土堂町10番地 オムロン株式会社内

(72) 発明者 門脇 正規

京都府京都市右京区花園土堂町10番地 オムロン株式会社内

(74) 代理人 100092598

弁理士 松井 伸一

Fターム(参考) 5K032 BA11 DB19

5K048 AA00 BA25 DA02 DC04 EB02

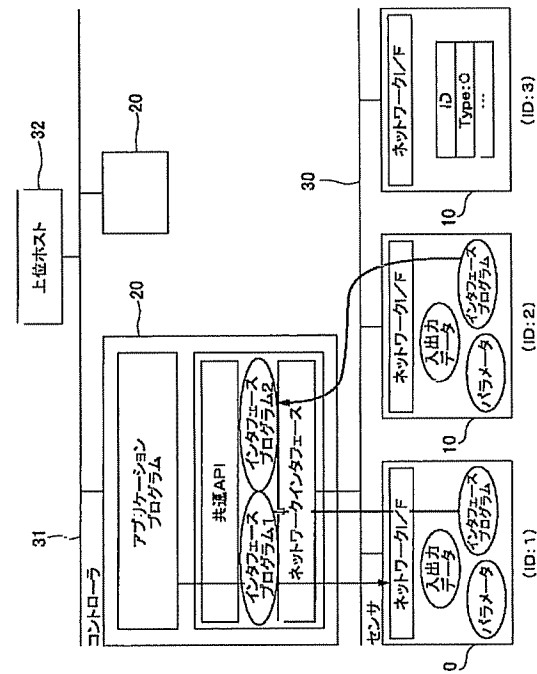
EB06 EB10 FC01 HA01 HA02

(54) 【発明の名称】 センサ及びコントローラ並びにセンサシステム

(57) 【要約】

【課題】 センサとコントローラ間で通信を行うためのセンサ固有の情報をコントローラにセットすることのできるセンサシステムを提供すること

【解決手段】 センサ10は、自己用のインタフェースプログラムを記憶しており、そのインタフェースプログラムは、ネットワーク30を介してコントローラ20に与えられる。コントローラは、センサから送られてきたそのセンサ用のインタフェースプログラムをアップロードし、その取得したインタフェースプログラムを用いて最適な動作環境でセンサにアクセスする。しかも、係るインタフェースプログラムは、ネットワークを介してダウンロードすればよいので簡単に行える。



## 【特許請求の範囲】

【請求項1】 ネットワークに接続取り外し可能なセンサであって、

自己用のインタフェースプログラムを記憶する記憶手段と、

前記記憶手段に格納されたインタフェースプログラムを前記ネットワークに出力する機能を備えたことを特徴とするセンサ。

【請求項2】 ネットワークに接続取り外し可能なコントローラであって、

前記ネットワークを介してセンサから送られてきたそのセンサ用のインタフェースプログラムをアップロードする機能と、

前記アップロードしたインタフェースプログラムを実行し前記センサにアクセスする機能を備えたことを特徴とするコントローラ。

【請求項3】 請求項1に記載のセンサと、請求項2に記載のコントローラをネットワークを介して接続し、前記コントローラは、前記センサ用のインタフェースプログラムをそのセンサからアップロードして取得し、その取得したインタフェースプログラムを用いて前記センサにアクセスするように構成したセンサシステム。

## 【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、センサ及びコントローラ並びにセンサシステムに関するものである。

【0002】

【従来の技術】従来、FAシステム等において各種の制御をする場合、PLCにセンサ等を接続し、そのセンサで検出した情報をPLCで解析し、その解析結果に基づいてPLCの制御対象の機器に対して制御命令を送り、協調・同期制御等を行うようになっている。

【0003】そして、近年、センサに信号処理機能を付加したものが開発されている。すなわち、センサをデバイスネットを介してコントローラの下位に接続する。そして、このコントローラがイーサネット等のネットワークを介して制御対象機器等と接続され、センサのセンシング情報に基づいてその制御対象機器に対してネットワークを経由して与えるようにしている。

【0004】係る構成において、コントローラのアプリケーションプログラムがセンサのデータや設定パラメータにアクセスする場合に、アプリケーションプログラムがセンサの物理アドレス（データポートのアドレス等）を意識する必要がある。しかも、係るアクセスするためのドライバは、アクセスモデル手順や、センサのレジスタモデル、機能内容に応じて設定する必要があり、センサの高機能化に伴い、実際には各センサ固有のドライバが必要となる。それに伴い、アプリケーションプログラム内に接続する各センサのドライバ（デバイス固有情報）を用意し、組み込んでおくことになる。

【0005】

【発明が解決しようとする課題】しかしながら、センサとコントローラの組み合わせは無数にあり、全てのセンサ用のドライバをコントローラ側にあらかじめ組み込んでおくことは困難である。一方、パソコンなどの世界では、外部装置を接続するに際し、その外部装置専用のドライバが必要な場合には、当該ドライバを記憶したFD等の記憶媒体を添付し、そのFD等をパソコンに装着してインストールすることにより対応できる。しかし、PLCその他のセンサにネットワークを介して接続されるコントローラは、多くの場合記憶媒体を読み取るためのドライブ装置はなく、そのまま転用することはできない。

【0006】また、例えばネットワーク上にパソコン等の記憶媒体を読み取ることのできるドライブ装置を有した上位ホストを接続し、一旦上位ホストに上記ドライバを記憶した記録媒体を介してセンサ固有のドライバ（デバイス固有情報）を記憶させ、当該ドライバをネットワークを介してコントローラにダウンロードすることも考えられる。

【0007】しかし、センサの交換の都度係る処理を行うのは非常に煩雑となる。しかも、センサ、コントローラと、上位ホストの設置位置は、離れていることが多いので、各場所に移動しつつそれぞれ所定の処理を行うのはより煩雑となる。

【0008】本発明は、上記した背景に鑑みてなされたもので、その目的とするところは、上記した問題を解決し、センサとコントローラをネットワークを介して接続することにより、簡単にそのセンサとコントローラ間で通信を行うためのセンサ固有の情報をコントローラにセットすることのできるセンサ及びコントローラ並びにセンサシステムを提供することにある。

【0009】

【課題を解決するための手段】上記した目的を達成するために、本発明に係るセンサでは、ネットワークに接続取り外し可能なセンサであって、自己用のインタフェースプログラムを記憶する記憶手段と、前記記憶手段に格納されたインタフェースプログラムを前記ネットワークに出力する機能を備えるように構成した（請求項1）。

【0010】また、本発明に係るコントローラでは、ネットワークに接続取り外し可能なコントローラであって、前記ネットワークを介してセンサから送られてきたそのセンサ用のインタフェースプログラムをアップロードする機能と、前記アップロードしたインタフェースプログラムを実行し前記センサにアクセスする機能を備えるように構成した（請求項2）。

【0011】さらに、本発明に係るセンサシステムでは、請求項1に記載のセンサと、請求項2に記載のコントローラをネットワークを介して接続し、前記コントローラは、前記センサ用のインタフェースプログラムをそ

のセンサからアップロードして取得し、その取得したインタフェースプログラムを用いて前記センサにアクセスするように構成した(請求項3)。

【0012】本発明によれば、センサの種類によって異なるインタフェースプログラムを、各センサ自体で保有し、そのインタフェースプログラムをコントローラのアップロード機能により、当該インタフェースプログラムをコントローラが取得する。したがって、以後の実際の運用では、アップロードしたインタフェースプログラムを用いてコントローラはセンサにアクセスできる。

【0013】つまり、コントローラはあらかじめ多数のセンサに合わせた固有のインタフェースプログラムに関する情報をもっていなくても、実際にセンサシステムを構築する際に、その構築するセンサからインタフェースプログラムを取得できるので、各センサにあった最適な環境で動作可能となる。

【0014】

【発明の実施の形態】図1は、本発明の第1の実施の形態を示している。同図に示すように、ネットワーク30に接続されて双方向に通信できるセンサ10と、ネットワーク(デバイスネット)30を通じてセンサ10と通信できるコントローラ20とにより、センサシステムが構築されている。このコントローラ20は、さらにイーサネット等のネットワーク31に接続され、他のコントローラや制御対象機器並びに上位ホスト32などの各種装置とネットワーク接続されるようになっている。

【0015】そして、概略を説明すると、各センサ10は、コントローラ20が汎用的にアクセスできるネットワークインタフェース(通信制御部)16を持ち、さらに、センサ固有のアクセス手順をコントローラ上で実行可能なプログラム(インタフェースプログラム)を備えている。

【0016】そして、各センサ10とコントローラ20がネットワーク30を介して接続された状態において、実際に運用を開始する前に、各センサ10がコントローラ20からの要求を受けて自己が格納するインタフェースプログラムをコントローラ20にアップロードする。これにより、コントローラ20はアップロードされたインタフェースプログラムを用いてセンサ10に対して通常のセンシング処理をするために必要なデータの送受が行えるようになる。

【0017】なお、開始当初にインタフェースプログラムをアップロードするために、センサ10とコントローラ20は、それぞれ汎用アクセスのためのネットワークインタフェースを有している。このネットワークインタフェースは、ネットワーク30に対して接続し、データの送受を行うことのできる基本機能を備えたインタフェースである。そして、上記した処理を実行するための本発明に係るセンサやコントローラの具体的な構造は以下のようにになっている。

【0018】本形態におけるセンサ10は、図2に示すように、下位にセンサヘッドが接続されるようになっており、このセンサヘッドから与えられる生データ(アナログ信号)は、一旦センサ10内に取り込まれ、センシング信号処理して得られたデータをネットワーク30を介してコントローラ20に送るようになっている。

【0019】すなわち、まず、センサヘッド11からのアナログ信号を受信する信号入力部12を有する。この信号入力部12は、A/D変換器12aを備え、アナログ信号をデジタル信号に変換する。またこのA/D変換器12aは、コンフィグレーションを実行し感度調整部12bで設定された感度で変換する。

【0020】また、信号入力部12(A/D変換器12a)の出力は、信号処理部14に与えられるようになっている。この信号処理部14は、センサヘッドで検出され、信号入力部12でデジタル化されたデータに対し、所定の信号処理(センシング信号処理)を行う。そして、係る処理は、信号処理回路14aが作業用記憶領域14b(RAM)を使用しながら実行し、その実行するためのプログラムはプログラム格納領域14cに格納されている。

【0021】この信号処理回路14aは用途に応じて例えば16ビットマイコン等で構成することができる。そして、信号処理(プログラム)はセンシング対象に応じて異なるとともに、従来公知のものをを用いることができるので、その詳細な説明を省略する。

【0022】そして、この信号処理部14から出力されるセンシング信号処理済みデータは、通信制御部16のデータ送信回路16aに与えられ、ネットワーク30を介して上位装置(コントローラ20)に送るようになっている。さらに、通信制御部16は、データ受信回路16bを備え、コントローラ20から送られる各種情報を受信可能としている。この通信制御部16が、ネットワークインタフェースの機能を有している。

【0023】さらに、本発明では、インタフェース格納領域17を有し、センサ固有のインタフェースプログラム、つまり、センサ固有のアクセス手順をコントローラ上で実行可能なプログラムが格納されている。そして、このインタフェース格納領域17は、通信制御部16と接続され、そこに格納されたインタフェースプログラムが通信制御部16を介してネットワーク30においてはコントローラ20に対して送信可能となっている。

【0024】一方、コントローラ20は、Javaのようなプラットフォーム非依存のプログラム実行環境を持たせている。そして、図3に示すように、従来と同様にアプリケーションプログラム実行部21や、ネットワーク30に接続し、データの送受を行う汎用アクセスのためのネットワークインタフェース(共通インタフェース)24を有している。

【0025】また、この実行部21でアプリケーション

プログラムを実行中にセンサ10に対して当該センサのデータや設定パラメータにアクセスする場合に必要とするセンサの物理アドレス（データポートのアドレス等）や実行手順等を記載したインタフェースプログラムを格納する格納領域22を有し、管理下におくセンサ固有のインタフェースプログラムをこの格納領域22内に格納するようになっている。

【0026】さらに、格納領域22に格納されたインタフェースプログラムとアプリケーションプログラムの間には、共通アプリケーションインタフェース（共通API）23を介在させている。この共通アプリケーションインタフェース23は、各センサ用のインタフェースプログラムの取り扱いの相違を吸収するので、アプリケーションプログラム側から見た場合に各センサ用のインタフェースプログラムが同じに見えるようにするためのインタフェースである。つまり、アプリケーションプログラムが共通の手順でセンサにアクセスするためのインタフェースである。

【0027】したがって、アプリケーションプログラムからは、インタフェースクラスで定義した共通アプリケーションプログラムを用いて、センサのインタフェースプログラムへアクセスできるようになる。これにより、アプリケーションプログラムは、センサの種類を意識することなく共通アプリケーションインタフェース23であらゆるセンサ10にアクセスできるようになる。

【0028】そして、この格納領域22に対するインタフェースプログラムのアップロード/ダウンロードは、システム構成管理部26の制御のもと、ロード制御部25により実行される。

【0029】そして、システム構成管理部26の具体的な機能は、図4に示すフローチャートのようにになっている。すなわち、まずネットワークに接続されたセンサ10に対して、センサIDのリードアクセスを発行（要求）する（ST1）。このリードアクセスを受信したセンサは、自己のセンサIDをコントローラに通知（応答）するので、その応答を受けたセンサIDからそのセンサが本発明対応か否か、つまり、自分用のインタフェースプログラムを有しているものか否かを判断する（ST2）。そして、本発明対応の場合には、ロード制御部25を動作させ、インタフェースプログラムをアップロードし格納領域22に格納する（ST3）。

【0030】次いで、アップロードしたそのセンサ用のインタフェースプログラムにしたがい、センサ10に対して初期化要求し（ST5）、センサから初期化完了通知を受け取ったならば、通常のセンシング処理に基づく運用を開始する（ST6）。なお、ステップ2の判断で、本発明対応のセンサでないとは判断された場合には、センサ固有情報をマニュアル操作で登録し（ST4）、その後その登録した情報にしたがってセンサ初期化要求をする（ST5）。このようにステップ2の判断を設け

たのは、ネットワーク30に接続するセンサは多岐にわたるため、必ずしもインタフェースプログラムを備えた本発明対応のもののみが接続されるとは限らないため、係る対応していないものが存在しても問題がない（システムが機能する）ようにするためである。

【0031】一方、センサ10側の機能は、図5に示すフローチャートのように動作する。すなわち、コントローラ20からセンサIDのリードアクセスを受信したならば、それに応答する。つまり、自己のIDをコントローラ20に対して送信する（ST11）。次いで、コントローラ20からの要求を待って、インタフェースプログラム格納領域17に格納された自己用のインタフェースプログラムをネットワーク30を介してコントローラ20に対してアップロードする（ST12）。

【0032】さらに、そのアップロードが完了するとコントローラから初期化要求があるので、その要求にしたがい、センサの初期化処理をする（ST13）。すなわち、コンフィグレーションデータのパラメータ調整を行い、最適な値で設定する。具体的には、A/D変換部12aの感度調整を行ったり、信号処理回路14aで行う信号処理、つまり判定閾値などのチューニングパラメータ、良品データ分布や不良品データ分布といった統計データ、入力サンプリング周期や出力フォーマットといった動作設定情報等を調整する。そして、係るセンサ初期化が終了したならば、その旨通知を発し、実際の運用（初期化された状態でセンシング処理実行）を行う（ST14）。

【0033】係る構成にしたため、センサ10とコントローラ20のコネクション確立時に、コントローラ20は、ネットワークインタフェース24を介して、接続されたセンサ10から固有のインタフェースプログラムをアップロードするとともに、自らのインタフェースブロック（格納領域22）にそれを組み込み、起動することができる。

【0034】よって、コントローラ20には、通信対象のセンサ10のインタフェースプログラムを取得できるので、コントローラ20は、接続されるセンサ10の種類を意識することなく、また特別なデータベースを持つ必要なく、あらゆるセンサに最適なインタフェース機能を持つことができる。

【0035】また、センサ10は、自己に適したインタフェースプログラムを保有し、それをコントローラ20にアップロードすることにより、当該インタフェースプログラムを用いてデータの送受等が行え、特別なリソースや取り決めを必要とせずに、あらゆるコントローラから最適なインタフェースでアクセスを受けることができる。

【0036】図6～図8は、本発明の第2の実施の形態の要部を示している。本実施の形態は、第1の実施の形態のより具体的な実施例の1つとも言える。まず、セン

サ10は、自己のレジスタモデルやアクセス手順等、固有のアクセス情報を実装したインタフェースプログラム（ドライバプログラム）をJavaプログラムとしてインタフェースプログラム格納領域17に保持している。

【0037】ここで、センサに格納するインタフェースプログラムは、コントローラ20の共通アプリケーション

・コンストラクタ

プログラムとして実体化し、センサに対する初期化を行う。

・デストラクタ

プログラムの実体を破棄し、センサに対する後処理を行ってセンサへのアクセスを不可とする。

・汎用手続き呼び出し

センサに対する詳細なコマンド並びにパラメータを指定し、その処理結果を受け取る。

・センサに固有な処理

センサ固有の出力データ読み出し、閾値パラメータ設定等を行うためのアドレス、アクセス手順等を格納するもので、汎用手続き呼び出しを受けて実行する。

【0039】上記したように、コントローラ20は、インタフェースプログラムとして汎用的なフォーマットを共通アプリケーションインタフェースとして保有しておき、センサ10からアップロードしてデータ（「出力データ読み出し」、「閾値パラメータ設定」）でオーバーライドすることにより、各センサ固有のインタフェースプログラムを構成するようしている。

【0040】なお、図示していないが、コントローラ20は、図3と同様にシステム構成管理部26やロード制御部25なども有している。システム構成管理部26の具体的な処理機能としては、図7に示すようになっている。同図と図4を比較するとわかるように、基本的な機能は同様であり、ステップ3でインタフェースプログラムをセンサからアップロードした後の手順が異なる。

【0041】つまり、コントローラは、Javaプログラムを実行可能な環境を持っている。そして、Javaのクラスロード機能を用いて、センサのインタフェースプログラムをコントローラの管理下へアップロードした（ST3）後、「出力データ読み出し、閾値パラメータ設定」等を該当領域にオーバーライドする（ST7）。そして、コンストラクタを実行し、アップロードしたインタフェースプログラム（Javaのクラス）をインスタンス化（実体化）する（ST8）。

【0042】その後の処理は、第1の実施の形態と同様で、センサ初期要求後（ST5）、運用開始する（ST6）。そして、その運用処理は、例えばアプリケーションからは、インタフェースクラスで定義した上記共通APIを用いて、実体化したセンサのインタフェースプログラムへアクセスする（図8中 $\Phi$ ）。これを受け、Javaの継承機能により、オーバーライドされたセンサ固有の処理が呼び出され、実行される（図8中 $\Phi$ ）。

ンインタフェースを定義したインタフェースクラスの継承クラスとして作成するもので、以下のメソッドが定義される。

【0038】

【表1】

【0043】なお、本実施の形態を、第1の実施の形態の図3との対応をとると、図3中の格納領域22に格納された各インタフェースプログラムと共通アプリケーションインタフェース23が融合・一体化して本実施の形態（図6）における共通アプリケーションインタフェース23'となっている。しいて分ければ、主として「汎用手続き呼び出し」が第1の実施の形態における共通API23に対応し、「出力データ読み出し」や「閾値パラメータ設定」が各インタフェースプログラムに対応すると言える。

【0044】また、別の見方をすると、第1の実施の形態では、インタフェースプログラム全体をコントローラに格納するようにし、第2の実施の形態ではコントローラ20にはインタフェースプログラムの一種のテンプレートを用意しておき、センサからはその位置の一部をアップロードし、そのテンプレートの該当箇所にオーバーライドする（関数単位でおきかえる）ことにより、センサ用のインタフェースプログラムを生成すると言える。

【0045】図9～図12は、本発明の第3の実施の形態を示している。本実施の形態は、上記した第1、第2の実施の形態を基本とし、さらにシステム構成情報のローカル保存機能を持たせている。すなわち、センサ10の内蔵インタフェースプログラムをアップロードすることにより、コントローラ20はシステム構成に対応したインタフェースに最適化される。また、チューニング等の初期化設定により、センサは動作可能な状態にコンフィグレーションされる。係る点は、上記の各実施の形態で説明した通りである。

【0046】そして、本形態では、インタフェースプログラムや、コンフィグレーションにより最適化された状態（コンフィグレーションデータ）をそれぞれの記憶手

段に保存しておくことにより、同一システム構成での再立ち上げの際に、センサ初期化（チューニング等）をすることなく記憶した最適な状態に設定することができ、高速化、機器の移設や交換、異常発生時の原因究明等が容易になる。そして、係る処理を実施するための具体的な構成としては、以下のようになっている。

【0047】まず、センサ10は、図9に示すように、コンフィグレーションデータ格納領域18を設ける。そして、センサ初期化によりチューニング・パラメータ調整等を行い最適化したコンフィグレーションデータが作業用記憶領域14b中のパラメータ変数領域に設定され、以後の運用はその最適化された状態でセンサ10が動作する。そして、そのように最適化されたデータをコンフィグレーションデータ格納領域18に格納する。これにより、次の起動時の際に前回からシステム構成の変更がない場合、格納領域18に格納されているコンフィグレーションデータに対応するパラメータ変数領域にロードするだけで初期化が終了する。

【0048】また、コントローラ20は、図10に示すように、インタフェースプログラム格納領域27を設け、センサに対応して最適化された状態のインタフェースプログラムを保存する。これにより、起動時に前回のシステム構成から変更がない場合、最初から汎用の共通ではなくインタフェースプログラム格納領域27に格納された、最適化されたインタフェースプログラムでAPIを実体化することができる。

【0049】なお、図示の例ではインタフェースプログラム格納領域27をコントローラ20の内部に設けた例を示したが、本発明はこれに限ることはなく、コントローラ20に外部記憶装置を接続し、その外部記憶装置に当該インタフェースプログラムを格納するようにしてももちろん良い。さらに、いずれの場合も、インタフェースプログラムのみでなく、最適化されたコンフィグレーションデータも対して格納するようにしても良い。

【0050】そして、コントローラ20のシステム構成管理部26の機能は、図11に示すようなフローチャートとなっている。なお、前提として前回までの処理で使用していた最適化済みインタフェースプログラムは、インタフェースプログラム格納領域27に格納しているものとする。

【0051】まずセンサのIDのリードアクセスを要求し、センサからの応答を待つ（ST21）。そして、前回終了時からシステム構成の変更があったか否かを判断し（ST22）、変更が無い場合には、インタフェースプログラム格納領域27に格納している最適化済みインタフェースプログラムを格納領域22にロードし、最適化されたインタフェースプログラムでAPIを実体化する（ST23）。その後、ステップ28に進み、運用開始する。

【0052】一方、システム構成に変更があった場合に

は、ステップ22からステップ24に進み、そのセンサに対して構成変更通知を発し、相手からの確認通知を待つ（ST25）。そして、確認通知が来たならば、上記した各実施の形態と同様に、本発明対応のセンサか否かを判断（ST25）し、対応の場合にはインタフェースプログラムのアップロードを要求し、アップロードを実行する（ST26）。また、対応していないセンサの場合には、マニュアル操作でセンサ固有情報を登録する（ST29）。そして、いずれかの手法によりインタフェースプログラムの登録が完了したならば、センサに対して初期化要求し（ST27）、センサからの完了通知を待って、運用を開始することになる。

【0053】また、センサ側の機能は、図12に示すように、コントローラ20から受信したリードアクセスの応答として自己のIDを送出する。その後、コントローラ20から構成変更通知が送られてきたならば、アンサーバックを返す（ST32）。この通知の受信によりシステム構成の変更があったことがわかる。また、本形態では、図11のステップ22から24へ行くルートからわかるように、システム構成変更が無い場合には構成変更通知（ST24）は発せられない。したがって、センサ側でもこれを考慮し、IDを送出した後、一定時間経過してもコントローラから構成変更通知がこない場合にはシステム変更が無かったと判断し、ステップ32をスルーする。

【0054】そして、ステップ33では、システム構成に変更があったか否かを判断し、無かった場合には、ステップ34に進み、センサ10のコンフィグレーション格納領域18に格納されたコンフィグレーションデータに対応する変数領域にロードし（ST34）、運用を開始する（ST37）。

【0055】一方、システム構成の変更があった場合には、その後コントローラ20からアップロードの要求があるので、インタフェースプログラム格納領域17に格納されたインタフェースプログラムをアップロードする（ST35）。そして、そのアップロードが成功すると、次にセンサ初期化要求がされるので、それを受けてチューニング等のセンサ初期化処理を行い、完了したならば完了通知を送る（ST36）。その後運用開始するようになる（ST37）。

【0056】図13～図15は、本発明の第4の実施の形態を示している。本実施の形態では、センサの記憶領域再利用を図るようにしている。すなわち、上記した各実施の形態では、いずれもセンサ10にはインタフェースプログラム格納領域17を確保し、コントローラにアップロードした後もその格納領域17内にインタフェースプログラムは記憶された状態としている。

【0057】しかし、センサは小型な方が良いのは言うまでもなく、また、インタフェースプログラムは、一旦コントローラ20にアップロードしたならば、システム

構成が変更されない限り使用しない。したがって、そのインタフェースプログラムを格納する領域の記憶容量が使用できない容量として無駄となっている。

【0058】そこで、図13に示すように、各センサ10がコントローラ20にインタフェースプログラム(P)をアップロードしたならば、インタフェースプログラムが格納されていた記憶領域に、その後の初期化処理で得られたコンフィグレーションデータ(D)を記憶保持するようにしている。また、上記した第3の実施の形態と同様に、アップロードしたインタフェースプログラムを外部記憶装置27'に格納するようにしている。なお、この外部記憶装置27'は、直接またはネットワークを介して間接的にコントローラ20と接続することができる。

【0059】なお、センサ10側の構造としては、図14に示すように、インタフェースプログラムを格納する格納領域16'を解放する機能を設けるとともに、解放後その格納領域16'にコンフィグレーションデータを格納する機能を設けている。つまりセンサ10は、インタフェースプログラム格納領域をコンフィグレーションデータ格納用に共用する。そして、その他の構成は基本的に上記した各実施の形態と同様である。

【0060】なお、本形態の図示の例は、外部記憶装置27'を設けたが、第3の実施の形態と同様に内蔵させても良い。また、この例では外部記憶装置27'には、コンフィグレーションデータも一緒に記憶保持しているが、記憶させるのはインタフェースプログラムだけでももちろん良い。

【0061】これにより、システム構成が変更されない限り、再立ち上げの際に、センサ初期化(チューニング等)をすることなく記憶した最適な状態に設定することができる。

【0062】さらに、センサをシステムから取り外す場合には、事前にコントローラ側の外部記憶装置27'に保存されているインタフェースプログラムをセンサにダウンロードして戻すようにする。これにより、ネットワークから外れたセンサ10には、そのセンサ用のインタフェースプログラムを戻す。これを行わないと、センサにはインタフェースプログラムが存在しなくなってしまう。

【0063】そこで、センサ10をシステム、つまりネットワーク30から外す場合には、外気部記憶装置27'に格納したインタフェースプログラムをダウンロードし、元の状態に戻すようにしている。

【0064】そして、上記の処理を実行するためのコントローラ20のシステム構成管理部26の機能は、図15に示すフローチャートのようにになっている。すなわち、人手により与えられたセンサの取り外し処理コマンドを受信すると(ST41)、対応するセンサに対して動作停止指示を送る(ST42)。そして、そのセンサ

に付いての保存プログラムが有るか否かを判断(ST43)し、無い場合には異常通知をして終了する(ST47)。

【0065】一方、保存プログラム(外部記憶装置27'に格納したインタフェースプログラム)が存在する場合には、センサ側の準備がOKか否かを判断する(ST44)。具体的には、ステップ42で送信した動作停止指示を受けたセンサが、動作を停止すると、その旨通知が来るので当該通知を受けると準備OKと判断する。そして、保存してあるインタフェースプログラムをセンサに対してダウンロードし(ST45)、センサの取り外し処理完了通知を発行し(ST46)、一連の処理を終了する。

【0066】また、センサ10側の機能は、図16に示すフローチャートのように、コントローラ20から発せられた動作停止命令を受信すると(ST51)、動作を停止するとともに格納領域16'を解放し、その後完了通知をコントローラ20に送る。

【0067】その後、コントローラから自己のインタフェースプログラムをダウンロードし、上記解放した記憶領域16'にインタフェースプログラムを格納し、復旧処理を終了する(ST53)。

【0068】これにより、インターネットやフロッピーディスク等の外部記憶手段または通信手段によって、接続されているセンサのID情報(たとえば型式等)に応じた内蔵インタフェースプログラムを入手し、離脱するセンサにダウンロードして復旧できる。

【0069】

【発明の効果】以上のように、本発明では、センサ固有のインタフェースプログラムをセンサ自身で保有し、センサとコントローラをネットワークを介して接続する際などにそのセンサからコントローラに対して当該インタフェースプログラムをアップロードすることにより、そのコントローラは当該センサに対する最適な環境でアクセスすることができる。

【0070】しかも、実際に接続する際に係るアップロード等をすればよいので、もともと各センサについての情報をもたせる必要もなく、メモリ効率もよいばかりでなく、センサのバージョンアップや新たなセンサに対しても対応できる。さらに、インタフェースプログラムは、ネットワークを介してダウンロードすればよいので簡単に行える。

【図面の簡単な説明】

【図1】本発明の第1の実施の形態を示す図である。

【図2】第1の実施の形態のセンサの内部構造を示す図である。

【図3】第1の実施の形態のコントローラの内部構造を示す図である。

【図4】第1の実施の形態のコントローラのシステム構成管理部の機能を説明するフローチャートである。



【図5】第1の実施の形態のセンサの機能を説明するフローチャートである。

【図6】第2の実施の形態を示す図である。

【図7】第2の実施の形態に用いられるコントローラのシステム構成管理部の機能を説明するフローチャートである。

【図8】その作用を説明する図である。

【図9】第3の実施の形態に用いられるセンサを示す図である。

【図10】第3の実施の形態に用いられるコントローラを示す図である。

【図11】第4の実施の形態のコントローラのシステム構成管理部の機能を説明するフローチャートである。

【図12】第4の実施の形態のセンサの機能を説明するフローチャートである。

【図13】本発明の第5の実施の形態を示す図である。

【図14】第5の実施の形態のセンサの内部構造を示す図である。

【図15】第5の実施の形態のコントローラのシステム

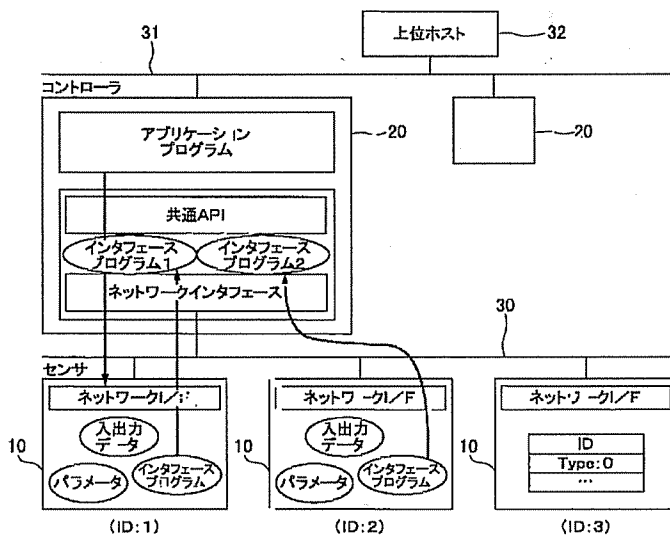
構成管理部の機能を説明するフローチャートである。

【図16】第5の実施の形態のセンサの機能を説明するフローチャートである。

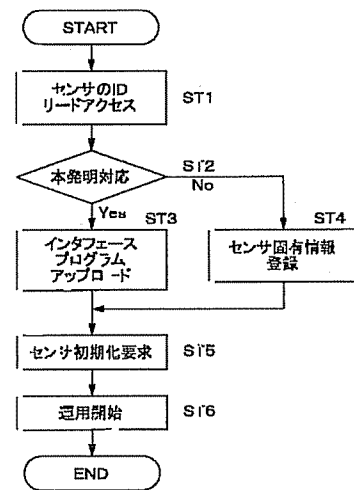
【符号の説明】

- 10 センサ
- 12 信号入力部
- 13 経路選択回路
- 14 信号処理部
- 16 通信制御部
- 17 インタフェースプログラム格納領域
- 18 コンフィグレーション格納部
- 20 コントローラ
- 21 アプリケーションプログラム実行部
- 22 格納領域
- 23 共通アプリケーションインタフェース
- 24 ネットワークインタフェース
- 25 ロード制御部
- 26 システム構成管理部
- 30 ネットワーク

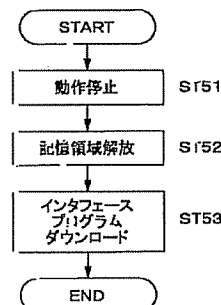
【図1】



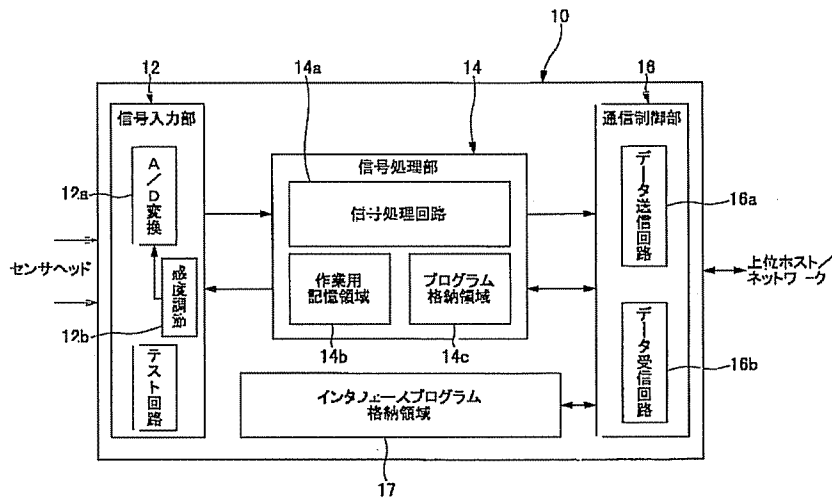
【図4】



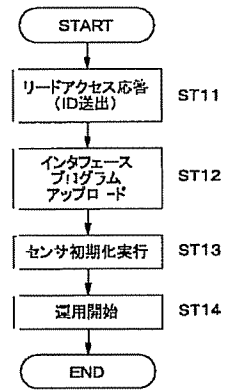
【図16】



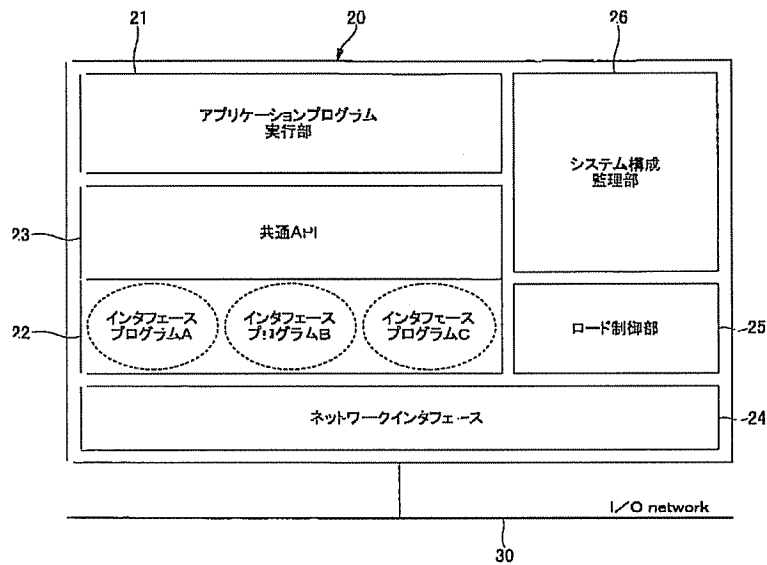
【図2】



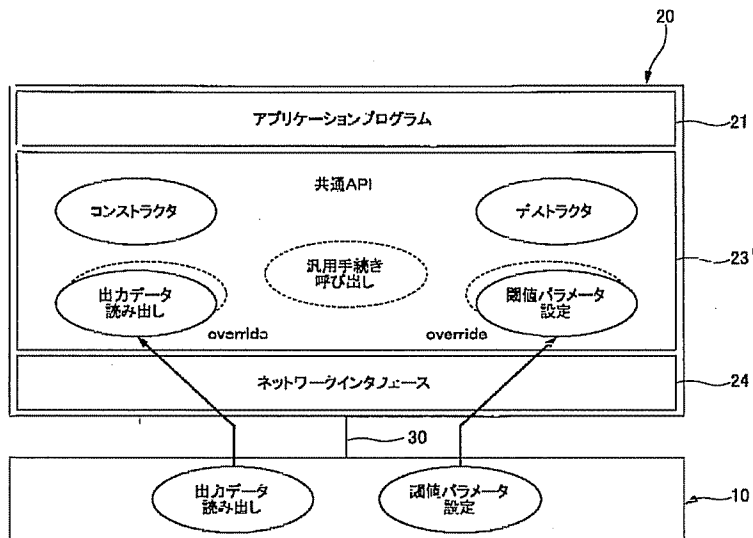
【図5】



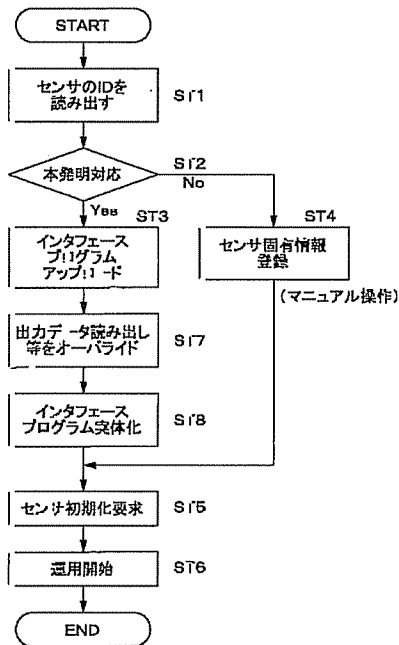
【図3】



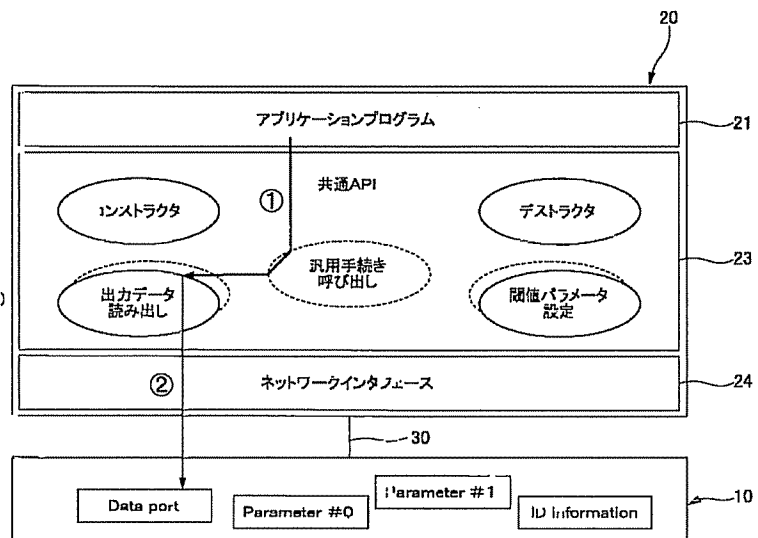
【図6】



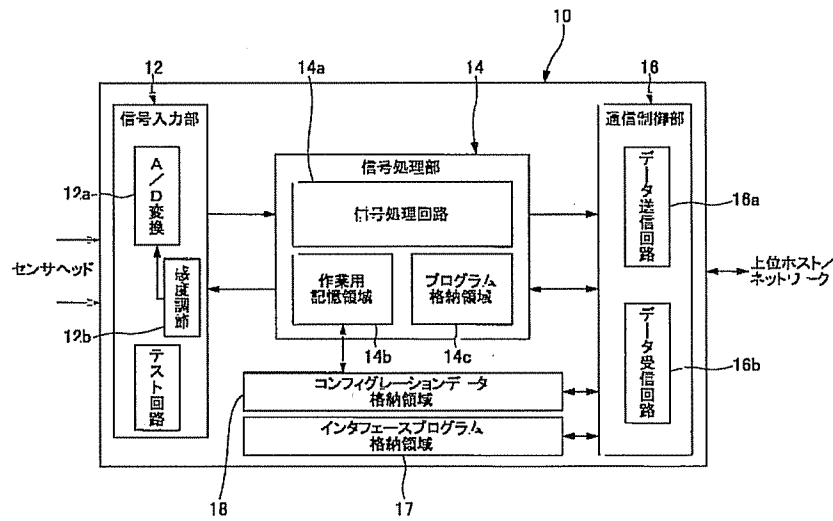
【図7】



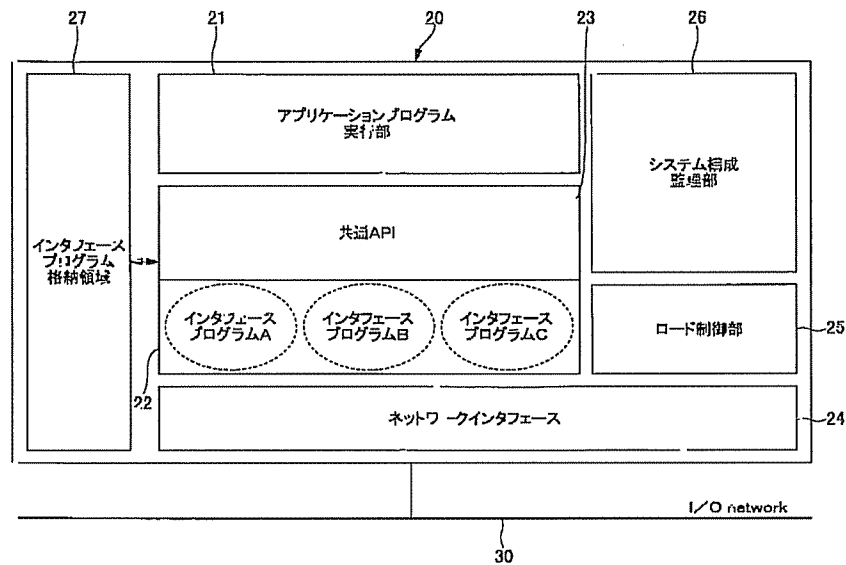
【図8】



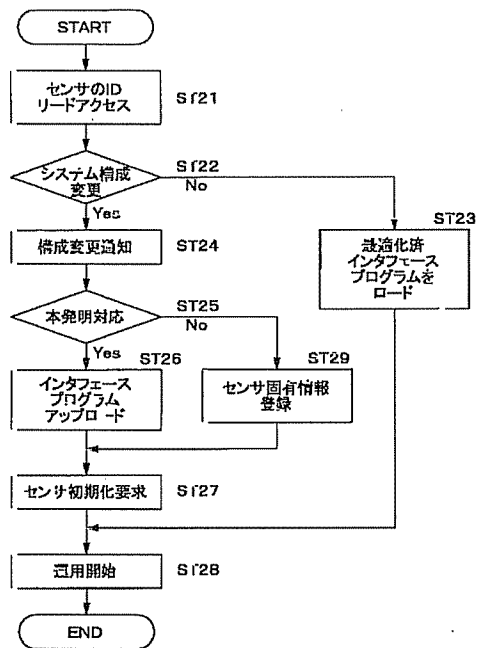
【図9】



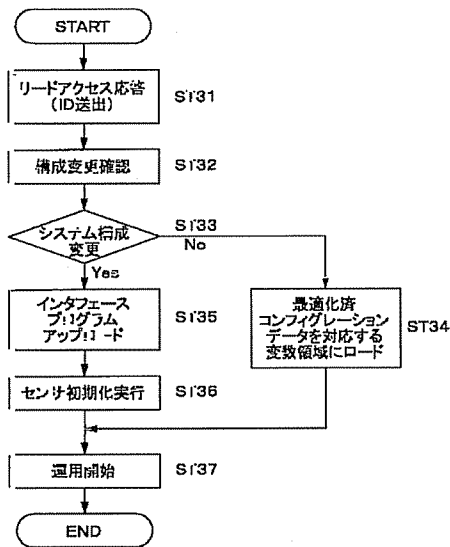
【図10】



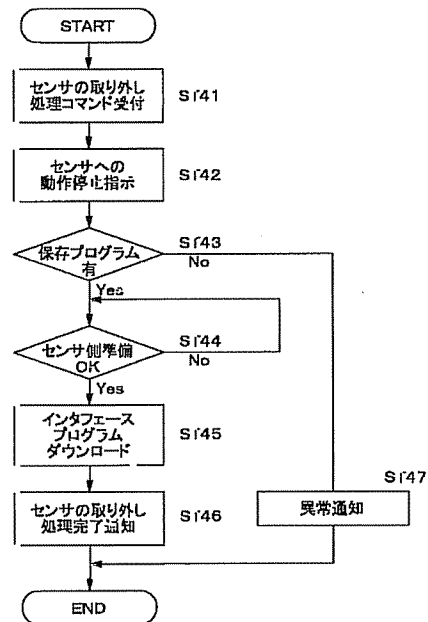
【図11】



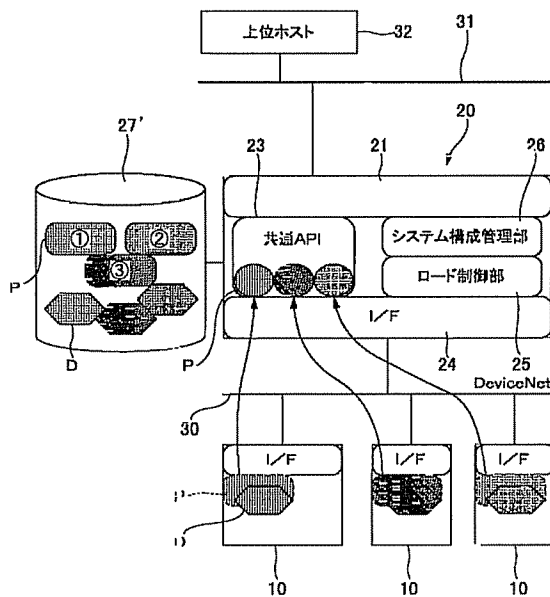
【図12】



【図15】



【図13】



【図14】

